МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

ИНСТИТУТ ТЕХНОЛОГИЙ ПРЕДПРИНИМАТЕЛЬСТВА

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| кандидат экономических наук |  |  |  | Е. И. Карасева |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| Жизненный цикл ИС |
|  |
| по дисциплине: Управление инновациями |
|  |

РАБОТУ ВЫПОЛНИЛА

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| СТУДЕНТКА ГР. № | 8726з | |  |  |  | Т.И. Березкина |
|  | |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

Модель жизненного цикла (ЖЦ) ИС — структура, определяющая последовательность выполнения и взаимосвязи процессов, действий и задач на протяжении жизненного цикла. Модель жизненного цикла зависит от специфики, масштаба и сложности проекта и специфики условий, в которых система создается и функционирует.

Модель ЖЦ ИС включает в себя:

• стадии;

• результаты выполнения работ на каждой стадии;

• ключевые события — точки завершения работ и принятия решений.

Модель жизненного цикла отражает различные состояния системы, начиная с момента возникновения необходимости в данной ИС и заканчивая моментом ее полного выхода из употребления.

Типы моделей жизненного цикла ИС В настоящее время наиболее известны и используются следующие модели жизненного цикла:

1. **Каскадная модель** (рис. 1) предусматривает последовательное выполнение всех этапов проекта в строго фиксированном порядке. Переход на следующий этап означает полное завершение работ на предыдущем этапе.

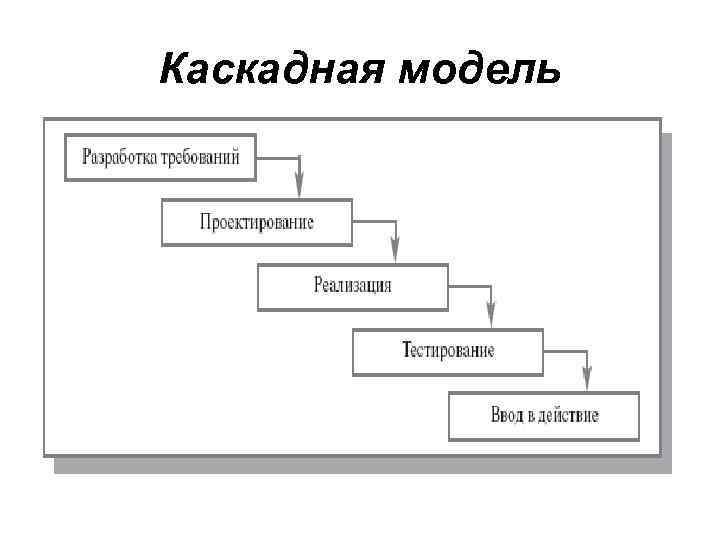


Рисунок 1 - Каскадная модель ЖЦ ИС

На первом этапе проводится исследование проблемы, которая должна быть решена, четко формулируются все требования заказчика. Результатом, получаемым на данном этапе, является техническое задание (разработка требований), согласованное со всеми заинтересованными сторонами.

На втором этапе разрабатывают проектные решения, удовлетворяющие всем требованиями, сформулированным в техническом задании. Результатом данного этапа является комплект проектной документации, содержащей все необходимые данные для реализации проекта.

Третий этап — реализация проекта. Здесь осуществляется разработка программного обеспечения (кодирование) в соответствии с проектными 2 решениями, полученными на предыдущем этапе. Методы, используемые для реализации, не имеют принципиального значения. Результатом выполнения данного этапа является готовый программный продукт.

На четвертом этапе проводится проверка (тестирования) полученного программного обеспечения на предмет соответствия требованиям, заявленным в техническом задании. Опытная эксплуатация позволяет выявить и исправить различного рода скрытые недостатки, проявляющиеся в реальных условиях работы информационной системы. Результат – готовая к эксплуатации система.

Последний этап — сдача готового проекта, ввод его в действие. Главная задача этого этапа — документально подтвердить, что все требования заказчика выполнены в полной мере. Результат – работающая ИС с полным комплектом сопроводительной документации, утвержденные заказчиком документы о принятии системы в эксплуатацию.

Можно выделить следующие **положительные стороны применения** каскадного подхода:

• на каждом этапе формируется законченный набор проектной документации, отвечающий критериям полноты и согласованности;

• выполняемые в логической последовательности этапы работ позволяют планировать сроки завершения всех работ и соответствующие затраты.

Каскадный подход хорошо зарекомендовал себя при построении относительно простых ИС, когда в самом начале разработки можно достаточно точно и полно сформулировать все требования к системе.

**Недостатки каскадной модели**. Перечень недостатков каскадной модели при ее использовании для разработки информационных систем достаточно обширен:

• существенная задержка в получении результатов;

• ошибки и недоработки на любом из этапов проявляются, как правило, на последующих этапах работ, что приводит к необходимости возврата назад;

• сложность параллельного ведения работ по проекту;

• чрезмерная информационная перенасыщенность каждого из этапов;

• сложность управления проектом;

• высокий уровень риска и ненадежность инвестиций.

Основным недостатком этого подхода является то, что реальный процесс создания системы никогда полностью не укладывается в такую жесткую схему, постоянно возникает потребность в возврате к предыдущим этапам и уточнении или пересмотре ранее принятых решений. Это приводит к существенной задержке в получении результатов.

В результате реальный процесс создания ИС оказывается соответствующим поэтапной модели с промежуточным контролем. Здесь разработка ИС ведется итерациями с циклами обратной связи между этапами. Межэтапные корректировки позволяют учитывать реально существующее взаимовлияние результатов 3 разработки на различных этапах; время жизни каждого из этапов растягивается на весь период разработки (рис. 2).



Рисунок 2 - Поэтапная модель с промежуточным контролем

1. **Спиральная модель** ЖЦ (рис. 3) была предложена для преодоления перечисленных проблем каскадной модели. На этапах анализа и проектирования реализуемость технических решений и степень удовлетворения потребностей заказчика проверяется путем создания прототипов.

Каждый виток спирали соответствует созданию работоспособного фрагмента или версии системы. Это позволяет уточнить требования, цели и характеристики проекта, определить качество разработки, спланировать работы следующего витка спирали. Таким образом, углубляются и последовательно конкретизируются детали проекта, и в результате выбирается обоснованный вариант, который удовлетворяет действительным требованиям заказчика и доводится до реализации.

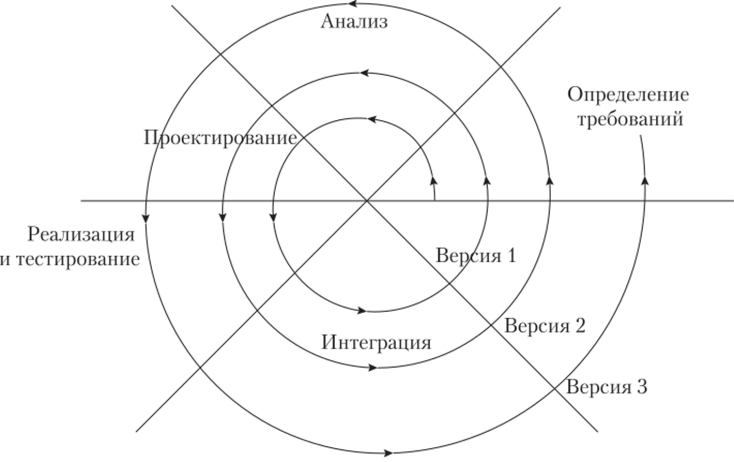


Рисунок 3 - Спиральная модель ЖЦ ИС

Использование спиральной модели позволяет перейти на следующий этап выполнения проекта, не дожидаясь полного завершения текущего — недоделанную работу можно будет выполнить на следующей итерации. Главная задача каждой итерации — как можно быстрее создать работоспособный продукт, который можно показать пользователям системы. Таким образом, существенно 4 упрощается процесс внесения уточнений и дополнений в проект.

Рассмотрим **преимущества итерационного** подхода, применяемого в спиральной модели, более подробно.

1. Существенно упрощается внесение изменений в проект при изменении требований заказчика.

2. При использовании спиральной модели отдельные элементы информационной системы интегрируются в единое целое постепенно. Поскольку интеграция начинается с меньшего количества элементов, то возникает гораздо меньше проблем при ее проведении (по некоторым оценкам, при использовании каскадной модели разработки интеграция занимает до 40 % всех затрат в конце проекта).

3. Уменьшение уровня рисков. Данное преимущество является следствием предыдущего, так как риски обнаруживаются именно во время интеграции. Поэтому уровень рисков максимален в начале разработки проекта. По мере продвижения разработки ожидаемый уровень рисков снижается. Данное утверждение справедливо при любой модели разработки, однако при использовании спиральной модели снижение уровня рисков происходит с наибольшей скоростью. Это связано с тем, что при данном подходе интеграция выполняется уже на первой итерации, и на начальных витках спирали выявляются многие аспекты проекта, такие как пригодность используемых инструментальных средств и программного обеспечения, квалификация разработчиков.

На рисунке 4 приведены в сравнении графики зависимости уровня рисков от времени разработки для каскадного и итерационного подходов.

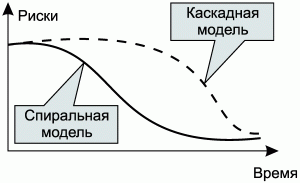


Рисунок 4 - Зависимость рисков от времени разработки

4. Итерационная разработка обеспечивает большую гибкость в управлении проектом, давая возможность внесения тактических изменений в разрабатываемое изделие. Например, можно сократить сроки разработки за счет снижения функциональности системы или использовать в качестве составных частей системы продукцию сторонних фирм вместо собственных разработок.

5. Итерационный подход спиральной модели ЖЦ упрощает повторное использование компонентов (реализует компонентный подход к программированию). Это обусловлено тем, что гораздо проще выявить (идентифицировать) общие части проекта, когда они уже частично разработаны, чем пытаться выделить их в самом начале проекта. Анализ проекта после проведения нескольких начальных итераций позволяет выявить общие многократно используемые компоненты, которые на последующих итерациях будут совершенствоваться.

6. Итерационный подход дает возможность совершенствовать процесс разработки — анализ, проводимый в конце каждой итерации, позволяет проводить оценку того, что должно быть изменено в организации разработки, и улучшить ее на следующей итерации.

**Недостатки спиральной модели**. Основная проблема спирального цикла — определение момента перехода на следующий этап. Для ее решения необходимо ввести временные ограничения на каждый из этапов жизненного цикла. Иначе процесс разработки может превратиться в бесконечное совершенствование уже сделанного. При итерационном подходе полезно следовать принципу «лучшее — враг хорошего». Поэтому завершение итерации необходимо проводить строго в соответствии с планом, даже если не вся запланированная работа закончена. Планирование работ обычно проводится на основе статистических данных, полученных в предыдущих проектах, и личного опыта разработчиков.

1. **Инкрементная модель** жизненного цикла (рис. 5) представляет собой пример итеративного подхода к разработке программного обеспечения ИС, который предполагает разбиение жизненного цикла проекта на последовательность итераций, каждая из которых напоминает «мини-проект», включающий все фазы жизненного цикла в применении к созданию отдельных версий системы, обладающих меньшей функциональностью по сравнению с проектом, в целом.

При этом на каждой итерации получается работающая версия программной системы, обладающая функциональностью всех предыдущих плюс текущей итерации. В результате финальной итерации получается конечный продукт, обеспечивающий реализацию всех требований. С точки зрения структуры жизненного цикла модель называют итеративной (говоря о процессе).

С точки зрения развития продукта – инкрементной (имеется ввиду наращивание функциональности продукта). Инкремент – приращение, увеличение (например, в языке программирование – увеличение значения переменной на 1).

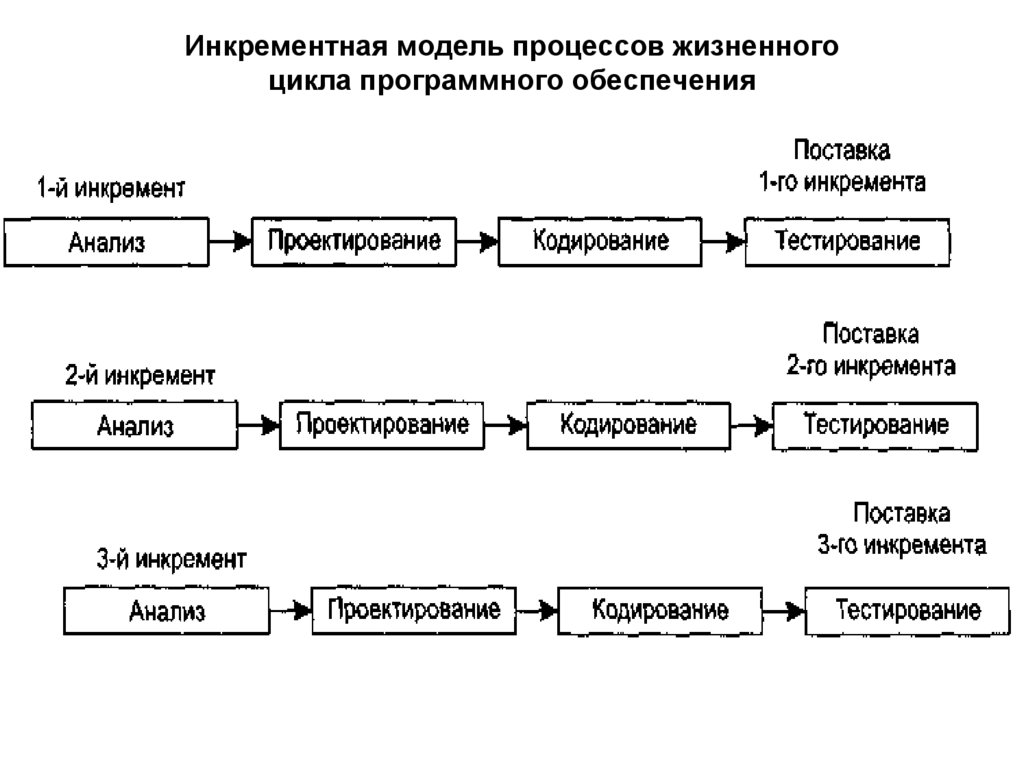


Рисунок 5 - Инкрементная модель жизненного цикла

Для каждого инкремента выполняется:

Анализ, на котором мы собираем требования и анализируем, и планируем сам инкремент;

• Проектирование, на котором происходит проектирование архитектуры, допроектирование тех вещей, которые не были сделаны на предыдущем инкременте;

• Разработка;

• Тестирование.

Важно, что каждый инкремент заканчивается работающим продуктом. Пусть он ограниченной функциональности, пусть у него не все реализовано, но это отдельный продукт, который можно показать, который можно отдать заказчику на тестирование.

Сам подход является однократным, т.е. мы весь наш проект делаем за один большой этап, имея в виду требования, но весь проект делится на инкременты, это небольшие версии продуктов, для которых заранее определена функциональность. Т.е. мы говорим, что у нас вся длительная разработка состоит, например, из 5 фаз. На первом этапе получим однопользовательскую версию, на втором инкременте (фазе) мы получим версию, которая поддерживает много пользователей, на третьем инкременте у нас появится поддержка веб-интерфейсов, а на четвертом какое-нибудь протоколирование и т.д. (рис. 6). Т.е. хоть требования у нас все равно задаются только один раз, но на выходе у нас появляется несколько инкрементов.

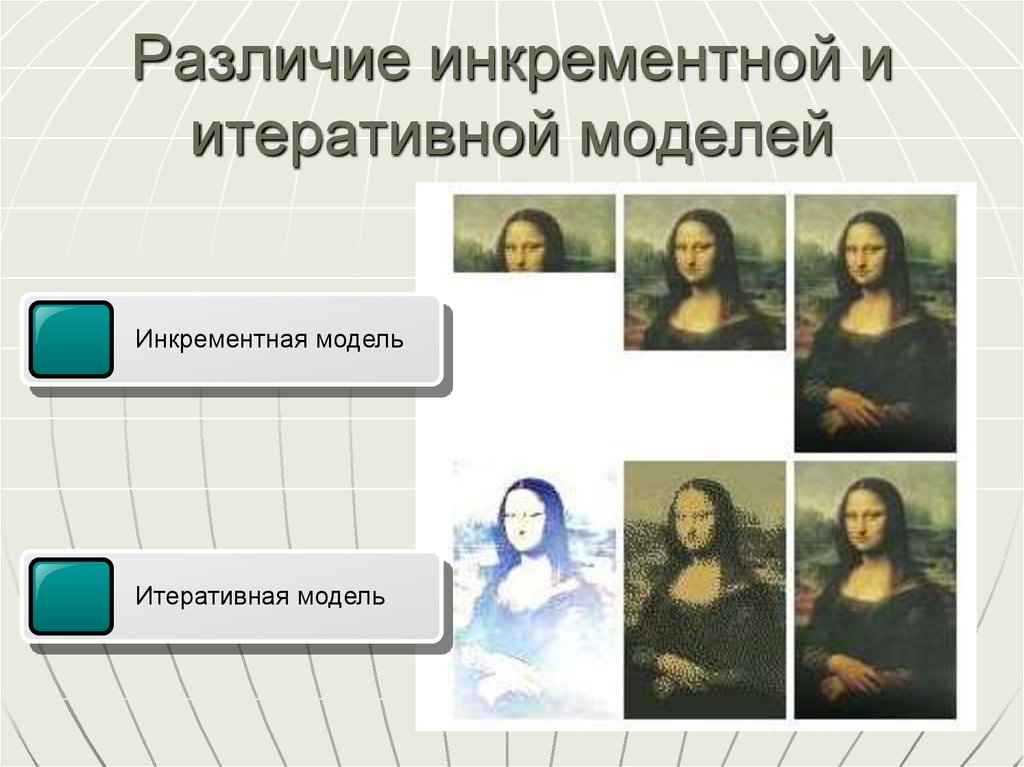


Рисунок 6 - Иллюстрация различий между инкрементной и итеративной (спиральной) моделями

**Преимущества инкрементной модели**:

• на момент создания определенного инкремента требования стабилизируются, поскольку не являющиеся особо важными изменения отодвигаются на момент создания последующих инкрементов;

• не требуется заранее тратить средства, необходимые для разработки всего проекта, поскольку сначала выполняется разработка и реализация основной функции или функции из группы высокого риска;

• в результате выполнения каждого инкремента получается функциональный продукт;

• ускоряется начальный график поставки (что позволяет соответствовать возросшим требованиям рынка);

• риск распределяется на несколько меньших по размеру инкрементов (не сосредоточен в одном большом проекте разработки);

• в конце каждой инкрементной поставки существует возможность пересмотреть риски, связанные с затратами и соблюдением установленного графика.

**Недостатки инкрементной модели**:

• в модели не предусмотрены итерации в рамках каждого инкремента;

• определение полной функциональности системы должно осуществляться в начале жизненного цикла, чтобы обеспечить определение инкрементов;

• формальный критический анализ и проверку намного труднее выполнить для инкрементов, чем для системы в целом;

• поскольку создание некоторых модулей будет завершено значительно раньше других, возникает необходимость в четко определенных интерфейсах;

• для модели необходимо хорошее планирование и проектирование;

• может возникнуть тенденция к оттягиванию решений трудных проблем на будущее с целью продемонстрировать руководству успех, достигнутый на ранних этапах разработки.